

NEFI: Network Extraction From Images

M. Dirnberger, T. Kehl and A. Neumann

Max Planck Institute for Informatics

Saarbrücken, Germany

February 19, 2015

Abstract

Networks and network-like structures are amongst the central building blocks of many technological and biological systems. Given a mathematical graph representation of a network, methods from graph theory enable a precise investigation of its properties. Software for the analysis of graphs is widely available [1–6] and has been applied to graphs describing large scale networks such as social networks, protein-interaction networks, etc. [7–10]. In these applications, graph acquisition, i.e., the extraction of a mathematical graph from a network, is relatively simple. However, for many network-like structures, e.g. leaf venations, slime molds and mud cracks, data collection relies on images where graph extraction requires domain-specific solutions or even manual approaches [11–14].

Here we introduce *Network Extraction From Images*, NEFI, a software tool that automatically extracts accurate graphs from images of a wide range of networks originating in various domains. While there is previous work on graph extraction from images, theoretical results are fully accessible only to an expert audience and ready-to-use implementations for non-experts are rarely available or insufficiently documented [15–19].

NEFI provides a novel platform allowing practitioners from many disciplines to easily extract graph representations from images by supplying flexible tools from image processing, computer vision and graph theory bundled in a convenient package. Thus, NEFI constitutes a scalable alternative to tedious and error-prone manual graph extraction and special purpose tools.

We anticipate NEFI to enable the collection of larger datasets by reducing the time spent on graph extraction. The analysis of these new datasets may open up the possibility to gain new insights into the structure and function of various types of networks. NEFI is open source and available at <http://nefi.mpi-inf.mpg.de>.

1 The Problem, Related Work and Motivation

The study of complex network-like objects is of increasing importance for many scientific domains. The mathematical study of networks, Graph Theory, formalizes a network’s structure by modeling the constituents of a network as *vertices*

and the pairwise relations between them as *edges*¹. Networks are ubiquitous in everyday life. Examples are as diverse as the Internet, social networks, transportation networks, metabolic networks, blood vessels or the vein networks of leaves. For a comprehensive review see [10].

In situations where the extraction of a mathematical graph from a physical network is easy, the size of graphs that can be analyzed quickly increased from hundreds to millions of vertices. At the same time it became feasible to build large databases of various types of networks. This enabled the application of software incorporating methods from statistics and graph theory to obtain many results that changed our understanding of large scale network structures. However, digitization remains difficult for many types of networks, e.g. leaf venations, and therefore ready-to-analyze datasets are often not available. In these cases, investigation on a larger scale requires tedious and error prone data acquisition.

In many experimental settings networks are initially available as images and it is necessary to extract the associated graphs from these images before any analysis can take place. This requires the identification of vertices and edges within the depicted structure. As this is a very work-intensive process, automated solutions are needed.

Leveraging advances in computer vision, several authors have proposed and successfully implemented solutions for domain specific graph extraction applications.

The authors of [15, 20] consider the mycelial networks of *P. impudicus*. They use watershed segmentation in combination with a novel enhancement step designed to highlight curvilinear features in the input networks. Based on the segmented image a skeleton is computed and used to extract the graph representing the input network. The resulting method is designed to be brightness and contrast invariant in order to correctly extract the networks grown by *P. impudicus* from challenging noisy or low contrast images.

Baumgarten et al. [13, 21] investigate the vein networks of *P. polycephalum*. For segmenting the input image they rely on careful constant thresholding followed by a sequence of restoration algorithms. Next, the restored segmented image is used to compute a skeleton. After applying another sequence of correction steps, the skeleton is scanned to extract the graph of the input network. The proposed approach is straight-forward and designed with a focus on images produced under controlled lab conditions.

In [16] a more general algorithm applicable for a variety of problems is proposed. Based on an original stochastic model, the authors use Monte Carlo sampling to obtain junction-points in the input image. This technically involved solution guarantees structural coherence for the resulting graph representation. Further examples include the extraction of road networks [18], retinal blood vessel analysis [17] and the extraction of plane graphs [19].

The above mentioned algorithmic solutions for the network extraction problem

¹Some communities traditionally refer to vertices as nodes or sites and to edges as arcs or links.

exhibit one or more of the following limitations:

- They do not build on top of well-established computer vision methods and tend to rely on ad-hoc algorithms. As a result the quality of the solution and its implementation could likely be improved. In addition, a lot of time is spent on reimplementing algorithms that are already available.
- They are not implemented or only available as pseudo-code.
- They are implemented but not designed for easy of use, distribution and extendability.

We are aware that the primary objective of the work cited above is not the production of reusable software, but of tools supporting a concrete research question. As a result, the above authors have limited time for researching the latest advances in computer vision, software engineering techniques or writing documentation. While we understand that under these circumstances the aforementioned limitations arise naturally, we strongly believe that it is necessary to overcome those limitations in order to increase the value and the impact of network extraction software. This has become the major motivation in developing NEFI.

Our goal is to enable virtually anyone to automatically extract networks from images. To this end we present an extensible framework of interchangeable algorithms accessible for the non-expert through an intuitive graphical user interface. Simultaneously, we envision NEFI as a flexible platform inviting experts in computer vision, image processing and software development to improve and extend NEFI's capabilities and thus promoting their own work to a wide interdisciplinary audience of users.

2 Network Extraction From Images

NEFI is a collection of image processing routines, segmentation methods and graph algorithms designed to process 2D digital images of various networks and network-like structures. Its main function is executing a so-called extraction pipeline, designed to analyze the structures depicted in the input image. An extraction pipeline, for short pipeline, denotes an ordered sequence of algorithms. A successful execution will return a representation of the network in terms of a weighted undirected planar graph. Computed weights include edge lengths and edge widths. Once the graph is obtained, available graph analysis software [1–6] or custom written scripts can be deployed to investigate its properties (see Supplementary Information).

A typical pipeline combines algorithms from up to four different classes: preprocessing, segmentation, graph detection and graph filtering, see Figure 1. For each class, NEFI typically offers several interchangeable algorithms to choose from. After executing preprocessing routines, a segmentation algorithm separates foreground from background. Then the foreground is thinned to a skeleton from

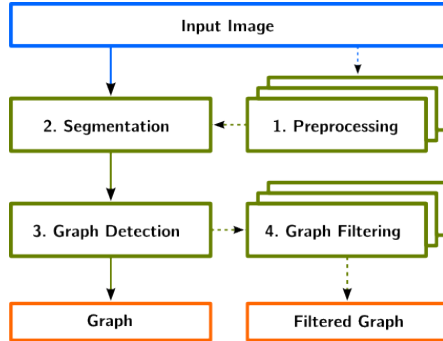


Figure 1: A flow chart illustrating NEFI’s pipeline components in green boxes. Dashed arrows depict optional sections of the pipeline. Blue and orange boxes denote NEFI’s input and outputs respectively.

which the vertices and edges of the graph are determined. In the process various edge weights are computed. Finally, the graph can be subjected to a variety of useful graph filters (see Supplementary Figure 4).

There are a number of predefined pipelines to get started immediately and with minimal effort. Alternatively, users may freely combine the various methods to build custom pipelines. Both approaches allow the user to experiment with the available methods in order to close in on the optimal settings for the data. Once a pipeline is constructed, it can be saved and reused. NEFI’s simple pipeline concept together with a self-explanatory graphical user interface make working with NEFI intuitive and straightforward (see Supplementary Figure 5). NEFI also offers a commandline mode, which is suited for batch processing.

NEFI comes with a number of example images from different domains which we use to produce the figures in this work. Figure 2 and Figure 3 show NEFI’s output on two example images using predefined pipelines. Blue squares denote the vertices and red lines the edges of the detected graph. The thickness of the detected edges corresponds to thickness of the depicted structures. For comparison the graph is drawn on top of the input image.

We stress that NEFI can deal with a range of inputs from various domains. In addition to the examples shown above, it has been successfully used to process images of natural (e.g. leaf venation, patterns of mud cracks) as well as man-made structures (tilings). It is also straightforward to add custom extensions. We provide a well documented framework that allows programmers to include more specialized segmentation algorithms or additional graph filters. For an overview of alternative graph extraction approaches see for example [22].

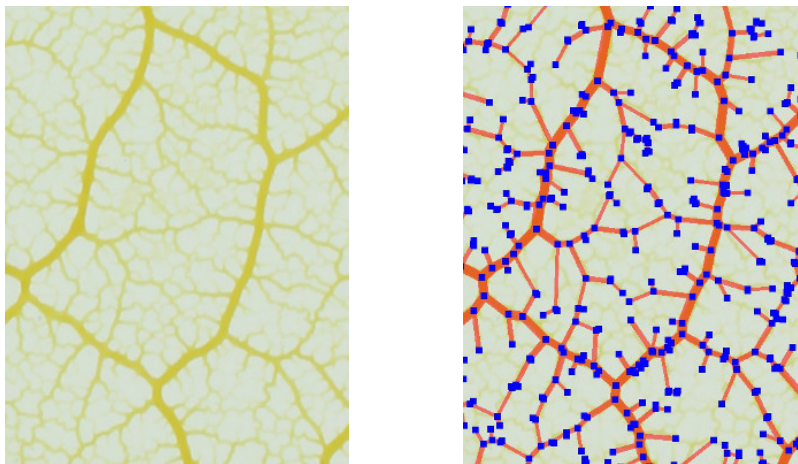


Figure 2: Extracted graph of the network formed by a slime mold (*Physarum polycephalum*). The left hand side shows the input image depicting the network. The right hand side shows the extracted graph overlayed on top off the same image for direct comparison. The image was produced in a collaboration with the KIST Europe.

3 Performance, Limitations and Comparison with similar Software

Depending on the input image and the pipeline, the quality of the resulting graph may vary. The major factor determining the quality of the extracted graph is the segmentation step. That is, if the image is segmented reasonably well, then the resulting graph can be expected to be accurate. In addition, graph filters can be deployed to remove stray vertices.

A quantitative assessment of the segmentation step is difficult as we are mainly interested in its influence on the quality of the detected graph. The problem is twofold: First, two different segmented images may lead to exactly the same graph. This becomes more likely after the application of graph filters. Second, a measure of the agreement between the depicted network and the extracted graph is not well-defined and best assessed via visual inspection. For this reason NEFI’s GUI allows quick comparison of input and output (see Supplementary Figure 5).

In contrast to other methods, we do not try to “repair” the segmented images or the skeleton using heuristics or user assisted methods. We thus retain a maximum of structural information which can be exploited by graph filters, allowing them to correctly remove artifacts after the graph has been established. For example, noisy regions in the input image may lead to a large number of spurious small connected components which can reliably be removed once the graph is available. Since the effects of direct manipulations on the graph itself

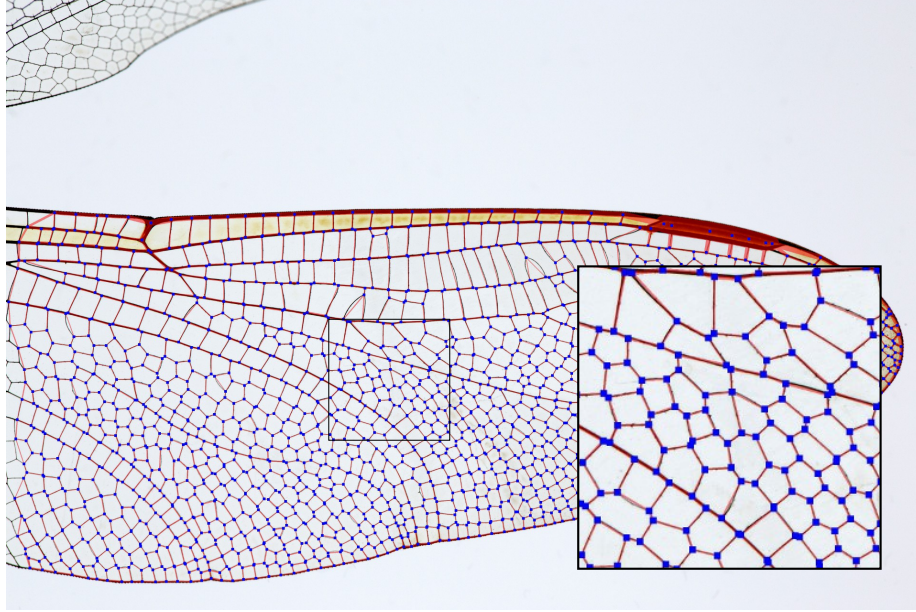


Figure 3: Extracted graph of the vein network exhibited by a wing of a dragonfly (*Ajax junius*). Image courtesy of Pam and Richard Winegar.

can immediately be evaluated by visual inspection, we prefer graph filtering over less transparent or more tedious approaches that take place before the graph was found (see Supplementary Figure 4).

NEFI was designed to process images produced under controlled laboratory conditions. Therefore, its general purpose methods do not work very well if the input contains irregular background or color/brightness gradients, has low contrast or insufficient resolution to detect structures that are either too dense or too fine. Nevertheless, NEFI aims to compensate for some of its limitations by offering the possibility to integrate additional algorithms with comparably little effort. NEFI should be regarded as a flexible platform suitable for further development rather than a universal solution to the network extraction problem.

Due to the nature of NEFI’s vertex detection, no vertices of degree two are detected. Furthermore, vertices of high degree (4 or more) are split into several degree three vertices. The latter can be merged by a suitable graph filter.

NEFI was designed to efficiently process large quantities of images. To this end it outsources much of its computationally intensive tasks to highly optimized and reliable libraries such as [OpenCV](#) [23] and [NetworkX](#) [24]. Table 1 illustrates the effectiveness of some of NEFI’s algorithms.

To assess NEFI’s value, we present a comparison with LEAF GUI [11], which we, to the best of our knowledge, consider our closest competitor in network extraction. LEAF GUI is a specialized MATLAB application geared towards

Pipeline element	Image small (1152×864)	Image large (5760×3840)
Watershed [25]	< 1	2
Adaptive Threshold	< 1	7
Guo-Hall Thinning [27]	< 1	12
vertex detection	< 1	5
Edge detection	< 1	6
Computing edge weights	< 1	5

Table 1: Timings of some of NEFI’s pipeline elements on images of different size. All values are in seconds. The timings were obtained on a Macbook Pro notebook equipped with a 2.4 GHz Intel i5 processor and 8 GB RAM.

investigation of leaf veins and areoles. The tool offers a comprehensive array of functionality accessible via a well-structured GUI. Although the usability and performance of LEAF GUI is good, NEFI implements a number of important improvements: NEFI improves on LEAF GUI by offering more sophisticated segmentation algorithms, i.e. guided Watershed [25] and GrabCut algorithms [26]. Additionally, NEFI implements the more reliable Guo-Hall thinning method [27], guaranteeing the connectivity of the skeleton. Since the segmentation and the resulting skeleton dominate the quality of the extracted graph, these improvements are critical. Furthermore, it becomes easier to use and compare different algorithms through the use of NEFI’s flexible pipeline concept. As a result the amount of user assistance NEFI requires to operate is much reduced when compared to LEAF GUI. Thus, given a suitable pipeline, batch processing of large amounts of images of comparable quality becomes a valuable option. Finally, NEFI’s source code is available for inspection and additional functionality can be added and then accessed via its streamlined GUI at any point. Table 2 summarizes the main results of our comparison.

4 Conclusion

We anticipate NEFI to become a valuable tool that allows scientists from any domain to automate graph extraction from images in an intuitive fashion requiring no expert knowledge. We hope that research scientists will be able to spend more time on analyzing their data and less time on processing it. By providing a flexible platform for graph extraction, we invite experts to extend and improve NEFI in order to introduce their contributions to a wider interdisciplinary audience. In the long run we would like NEFI to further the field of network analysis by promoting the creation of new network databases.

²While the documentation lists the possibility of computing the adjacency matrix which defines the graph, LEAF GUI V.1 does not allow the user to access this functionality via its GUI. We expect a future update to resolve this issue.

Features	NEFI	LEAF GUI
GUI	✓	✓
preprocessing	✓	✓
cropping	×	✓
segmentation	✓	✓
thinning	✓	✓
graph detection	✓	✓ ²
graph filtering	✓	×
graph attributes	✓	✓
predefined pipelines	✓	×
visual inspection of (intermediate) results	✓	✓
batch processing	✓	×
extensions possible	✓	×
general purpose	✓	×
free combination of algorithms	✓	×

Table 2: A comparison of basic features between NEFI 1.0 and LEAF GUI V.1.

References

1. Bastian, M., Heymann, S. & Jacomy, M. *Gephi: An Open Source Software for Exploring and Manipulating Networks* 2009.
2. Leskovec, J. & Sosič, R. *SNAP: A general purpose network analysis and graph mining library in C++* <http://snap.stanford.edu/snap>. June 2014.
3. Batagelj, V. & Mrvar, A. Pajek-program for large network analysis. *Connections* **21**, 47–57 (1998).
4. Xu, K., Tang, C., Tang, R., Ali, G. & Zhu, J. in *Communication Software and Networks, 2010. ICCSN '10. Second International Conference on* (2010), 350–354.
5. Loscalzo, S. & Yu, L. in *Social computing, behavioral modeling, and prediction* 151–159 (Springer, 2008).
6. Hagberg, A. A., Schult, D. A. & Swart, P. J. in *Proceedings of the 7th Python in Science Conference (SciPy2008)* (Pasadena, CA USA, Aug. 2008), 11–15.
7. Mislove, A., Marcon, M., Gummadi, K. P., Druschel, P. & Bhattacharjee, B. in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* (2007), 29–42.
8. D’Andrea, A., Ferri, F. & Grifoni, P. English. in *Computational Social Network Analysis* (eds Abraham, A., Hassanien, A.-E. & Snael, V.) 3–25 (Springer London, 2010).
9. Xenarios, I. *et al.* DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Research* **30**, 303–305 (2002).
10. Newman, M. The Structure and Function of Complex Networks. *SIAM Review* **45**, 167–256 (2003).
11. Price, C. A., Symonova, O., Mileyko, Y., Hilley, T. & Weitz, J. S. Leaf Extraction and Analysis Framework Graphical User Interface: Segmenting and Analyzing the Structure of Leaf Veins and Areoles. *Plant Physiology* **155**, 236–245 (2011).
12. Dhondt, S. *et al.* Quantitative analysis of venation patterns of Arabidopsis leaves by supervised image analysis. *The Plant Journal* **69**, 553–563 (2012).
13. Baumgarten, W. & Hauser, M. J. Detection, extraction, and analysis of the vein network. *Journal of Computational Interdisciplinary Sciences* **1**, 241–249 (2010).
14. Leung, K.-t. & Néda, Z. Criticality and pattern formation in fracture by residual stresses. *Phys. Rev. E* **82**, 046118 (4 2010).
15. Obara, B., Grau, V. & Fricker, M. D. A bioimage informatics approach to automatically extract complex fungal networks. *Bioinformatics* **28**, 2374–2381 (2012).

16. Chai, D., Forstner, W. & Lafarge, F. in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on* (2013), 1894–1901.
17. Krause, M., Alles, R. M., Burgeth, B. & Weickert, J. Fast retinal vessel analysis. *Journal of Real-Time Image Processing*, 1–10 (2013).
18. Negri, M., Gamba, P., Lisini, G. & Tupin, F. Junction-aware extraction and regularization of urban road networks in high-resolution SAR images. *Geoscience and Remote Sensing, IEEE Transactions on* **44**, 2962–2971 (2006).
19. Samuel, E., de la Higuera, C. & Janodet, J.-C. English. in *Structural, Syntactic, and Statistical Pattern Recognition* (eds Hancock, E. R., Wilson, R. C., Windeatt, T., Ulusoy, I. & Escolano, F.) 233–243 (Springer Berlin Heidelberg, 2010).
20. Obara, B., Frickerc, M. & Graua, V. in *SPIE Medical Imaging* (2012), 83141L–83141L.
21. Baumgarten, W. & Hauser, M. J. Computational algorithms for extraction and analysis of two-dimensional transportation networks. *J. Comput. Interdiscip. Sci* **3**, 107–16 (2012).
22. Dehkordi, M. T., Sadri, S. & Doosthoseini, A. A review of coronary vessel segmentation algorithms. *Journal of medical signals and sensors* **1**, 49 (2011).
23. Bradski, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).
24. Hagberg, A. A., Schult, D. A. & Swart, P. J. in *Proceedings of the 7th Python in Science Conference (SciPy2008)* (Pasadena, CA USA, Aug. 2008), 11–15.
25. Meyer, F. Un algorithme optimal pour la ligne de partage des eaux. *Dans 8me congrès de reconnaissance des formes et intelligence artificielle* **2**, 847–857 (1991).
26. Rother, C., Kolmogorov, V. & Blake, A. GrabCut - interactive foreground extraction using iterated graph cuts. *ACM Trans. Graphics (SIGGRAPH)*, 309–314 (2004).
27. Guo, Z. & Hall, R. W. Parallel thinning with two-subiteration algorithms. *Communications of the ACM* **32**, 359–373 (1989).
28. Bair, R. <http://www.tssphoto.com/>.

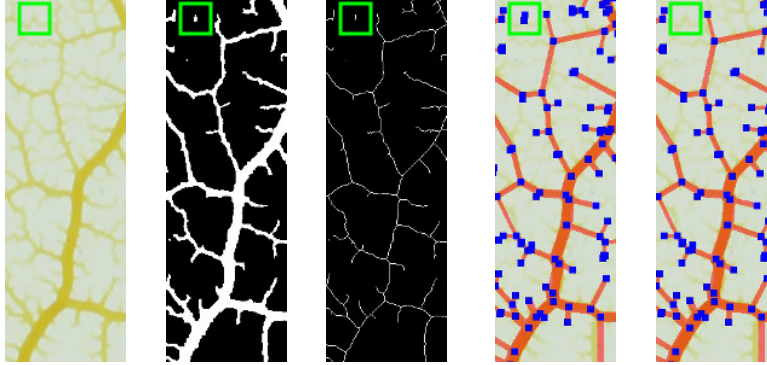


Figure 4: Direct comparison of NEFI’s pipeline steps given a slice of an image of a slime mold (*Physarum polycephalum*). From left to right: input image, segmented image, skeletonized image, detected graph and filtered graph. The green square contains a very faint vein which the segmentation does not pick up correctly. Thus, the skeleton becomes fragmented which leads to spurious vertices in the detected graph. By applying a graph filter we remove unwanted vertices without manipulation of the segmented or the skeletonized image. Similar filtering can remove "dead-ends", i.e. vertices that do not belong to any cycle in the graph.

A Supplementary information

A.1 General Information

NEFI is an open source Python application and available at <http://nefi.mpi-inf.mpg.de>. NEFI’s homepage includes a gallery of various use-cases and a comprehensive guide containing instructions on how to download, install and use the latest version of NEFI on Windows, Mac and Linux.

A.2 Pipeline and Graphical User Interface

Figure 4 shows the intermediate results of NEFI’s pipeline listed in the order of their execution. When a pipeline is executed, NEFI makes all intermediate results available via its clean and intuitive GUI, see Figure 5. Using the GUI all basic functions of NEFI can be accessed in an intuitive fashion

A.3 Comparison between NEFI and LEAF GUI

To assess the difference in quality of the respective output of NEFI and LEAF GUI one would like to compare the output graphs. However, since LEAF GUI currently does not make a graph representation available, we resort to a comparison of the segmentation and the thinning steps. This is justified, since these steps determine the quality of the extracted graphs.

For segmentation, LEAF GUI offers basic adaptive and constant thresholding. On top of LEAF GUI’s algorithms, NEFI’s segmentation additionally

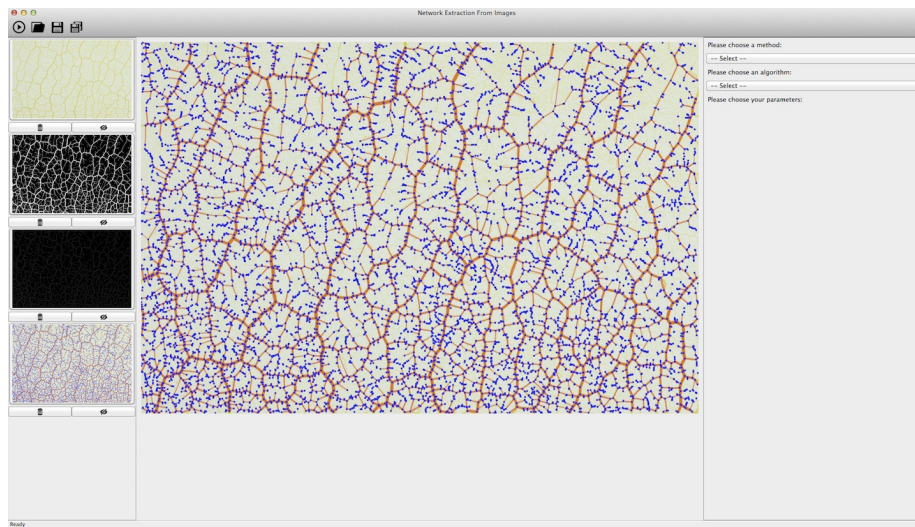


Figure 5: A screenshot of NEFI's GUI running on Mac OS. On the left hand side NEFI lists intermediate results as thumbnails. Bringing the final result to the center workspace allows for direct visual assessment of the quality of the extracted graph. On the right hand side NEFI's pipeline elements can be accessed.

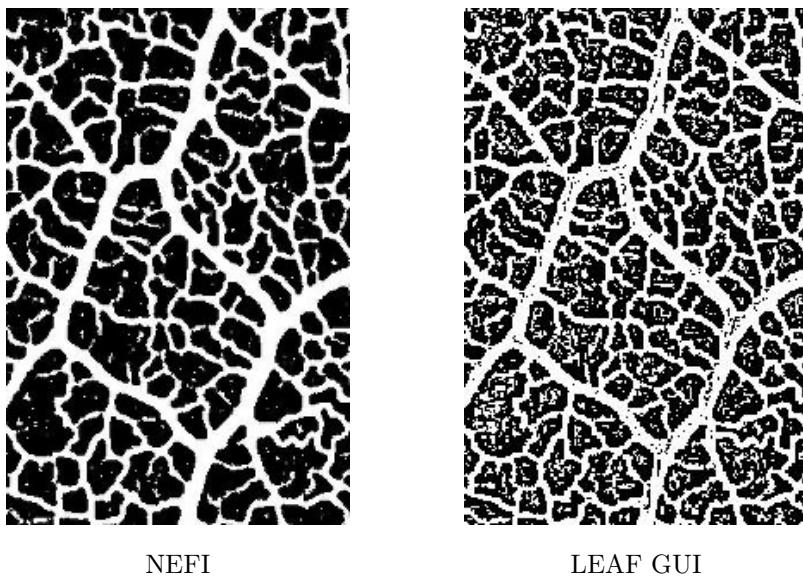


Figure 6: Comparison of segmented images of the network formed by a slime mold (*Physarum polycephalum*). Note how NEFI’s fully automatic Watershed based segmentation is sensitive and robust to noise at the same time. Both results may be improved further by careful manual tuning of the algorithm settings.

includes different variations of more advanced segmentation methods such as Watershed [25] or GrabCut [26] methods. Critically, for a general purpose tool, these allow for a wider range of images to be segmented correctly. Figure 6 and Figure 7 compare segmentation performance on two representative images.

For skeletonization, LEAF GUI uses an iterative thinning approach which tends to produce a highly fragmented skeleton. In contrast, NEFI utilizes the well-established method by Guo and Hall which preserves connectivity [27]. In many cases this algorithm produces a superior skeleton, allowing for robust graph detection. Figure 8 and Figure 9 show the skeleton images derived from the respective segmented images seen in Figures 6 and 7.

A.4 Analysis of Graphs

NEFI is a tool that facilitates data acquisition, which is a necessary precursor to data analysis. While NEFI provides users with valuable data in form of graphs, understanding this data is even more important. To analyze NEFI’s output one can either rely on open source graph analysis software [1–6] or write custom analysis programs dealing with special situations and computing non-standard observables. The latter requires some familiarity with software libraries such as Boost in C++ or NetworkX in Python [24], which are capable of dealing

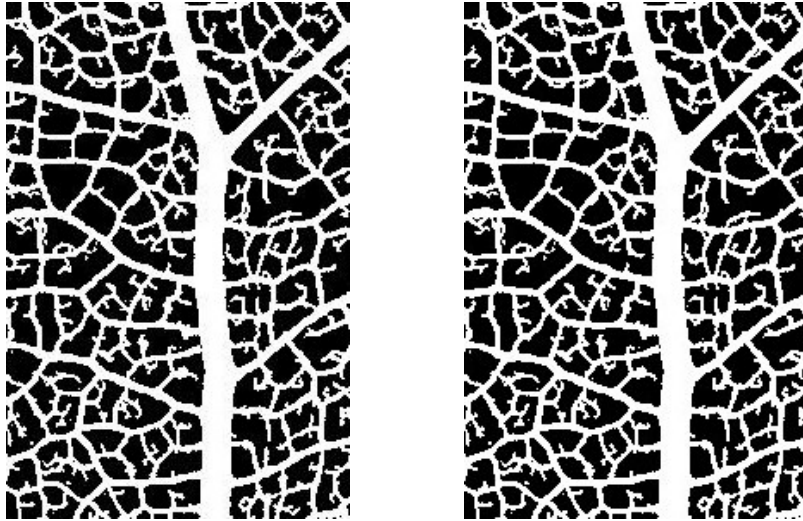
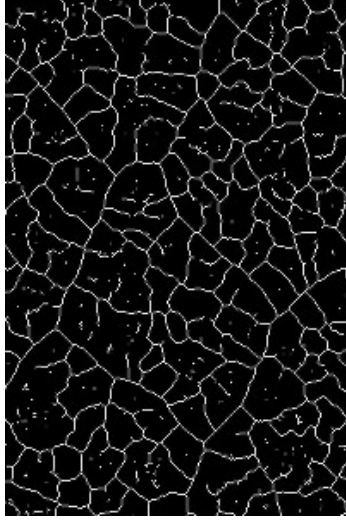
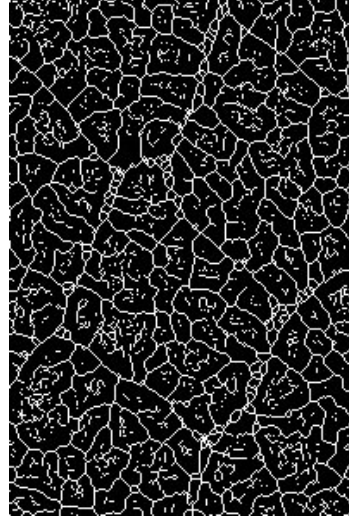


Figure 7: Comparison of segmented images of the vein network formed by a leaf (species unknown to the authors) [28]. The results are identical because the optimal segmentation algorithm was found to be adaptive thresholding, a method available both in NEFI as well as in LEAF GUI. The result indicates that LEAF GUI’s segmentation is competitive within its domain of use.

with graphs. To get the user started immediately, we provide a minimal Python program that illustrates the basic steps required to perform graph analysis. The code shows how to read a graph from disk given NEFI’s output and how to compute a histogram of a given edge attribute, e.g. edge length in pixel. The code can be downloaded from NEFI’s project page.

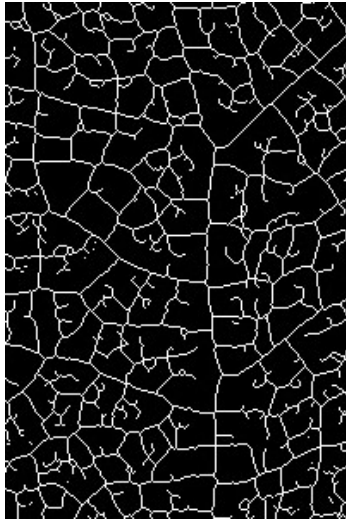


NEFI

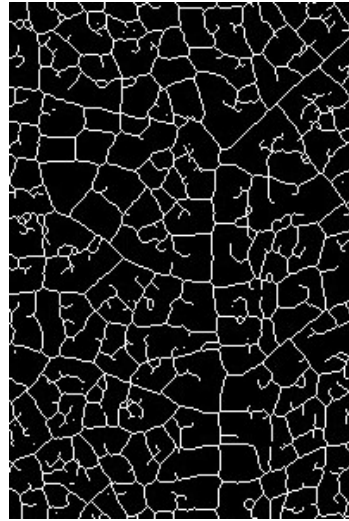


LEAF GUI

Figure 8: Comparison of skeletonized images of the network formed by a slime mold (*Physarum polycephalum*). Note how NEFI's Guo Hall thinning is much less likely to produce artifacts. LEAF GUI offers the possibility of repairing the skeleton manually.



NEFI



LEAF GUI

Figure 9: Comparison of segmented images of the vein network formed by a leaf (species unknown to the authors) [28]. Both thinning algorithms produce results of comparable quality.